Preliminary Study on Real-time Interactive Virtual Fixture Generation Method for Shared Teleoperation in Unstructured Environments

Vitalii Pruks[1] ✉, Ildar Farkhatdinov[2], and Jee-Hwan Ryu[1]

[1] Department of Mechanical Engineering, KOREATECH,
1600, Chungjeol-ro, Byeongcheon-myeon, Dongnam-gu, Cheonan-si,
Chungcheongnam-do, 31253, Republic of Korea
vprooks@koreatech.ac.kr, jhryu@kut.ac.kr
http://robot.kut.ac.kr
[2] School of Electronic Engineering and Computer Science,
Queen Mary University of London, UK
i.farkhatdinov@qmul.ac.uk

Abstract. We present a method for interactively generating virtual fixtures for shared teleoperation in unstructured remote environments. The proposed method allows a human operator to intuitively assign various types of virtual fixtures on-the-fly to provide virtual guidance forces helping the operator to accomplish a given task while minimizing the cognitive workload. The proposed method augments the visual feedback image from the slave's robot video camera with automatically extracted geometric features (shapes, surfaces, etc.) computed from both depth and color video sensor attached next to the slave robot's base. The human operator can select a feature on the computer screen which is then automatically associated with a virtual haptic fixture. The performance of the proposed method was evaluated with a peg-in-hole task and the experiment showed improvements in teleoperation performance.

Keywords: virtual fixture, bilateral teleoperation, unstructured environment, virtual fixture generation, computer vision.

1 Introduction

In teleoperation, it is well-known that virtual fixtures can assist human operators to improve task performance and safety [14, 11, 1, 10]. Generally, virtual fixtures can be of two categories: 1) guidance fixtures which provide assistive forces to guide a human operator to a target motion and 2) forbidden region virtual fixtures which define resistive force fields to prevent a human operator to work in certain areas of the workspace [2, 4].

Most of the research works on virtual fixtures assume that the fixtures are already present in the environment, or are associated with certain models in a structured environment. The pioneering works on virtual fixtures used material

fixtures made of plastic sheets [15, 14]. Such fixtures are installed at the master device and provide assistive guidance and forbidden region effects via direct contact with the master device. Assistance and avoidance fixtures are computed automatically with edge or blob detection algorithms in a sufficiently structured environment [3, 9]. An environment-specific knowledge embodied into the system allows generating virtual fixtures for more complex scenarios like surgery on a beating heart [16]. Computer vision algorithms using a 3D model [12] or a machine learning algorithm [18] to detect objects and their spatial poses in the visual scene can also be used for automatic fixture definition. In mobile robot teleoperation forbidden region virtual fixtures are defined with respect to the remote environment's objects and help to avoid collisions [6, 5]. But, it is not always possible to pre-define virtual fixtures, especially in unstructured and uncertain remote environments, which are the most common cases in teleoperation. Therefore, there has been a necessity of real-time virtual fixture generation method in teleoperation scenarios where no prior knowledge about the remote environment is given. For instance, in disaster scenarios, debris can have arbitrary shapes, and common equipment might be damaged and deformed. In addition, the method should also be intuitive and easy to use on-the-fly. Spending extra time on the generation of virtual fixtures might result in less benefit compared to direct teleoperation without the help of virtual fixtures.

Surprisingly, there has been not much discussion on generating virtual fixtures in real-time during the teleoperation. In [17] an initial trial for interactive virtual fixture generation was presented. The objects of the remote environment were defined as single points which enabled to associate them with virtual fixtures implemented with potential force fields. In [13], the authors proposed a method to specify a point-based path on a surface using a 2D image. In that approach, a human operator picked a set of multiple points on a surface shown from the remote video camera. These points were then used to define the path-based virtual fixtures for robot guidance. Both approaches presented in [17] and [13] were based on point-based interaction with the environment. Unlike in previous works, in the present paper we propose surface-based virtual fixture generation method. Our approach enables interactive and efficient virtual fixture assignment based on the geometrical shapes recognized from the three-dimensional image acquired from camera at the slave side. The proposed system includes a depth video camera at the slave side and a graphical user interface (GUI) at the master side. A 3D image from the slave robot's camera is used by a human operator to select the objects of interest and associate them with guidance or forbidden region virtual fixtures. These virtual fixtures can be defined and modified by a human operator interactively during the teleoperation.

The paper is organized as follows. The proposed virtual fixture generation method is described in details in section II. A user evaluation study with master-slave teleoperation system and its results are presented in sections III and IV. Discussion and conclusion can be found in the last section.

2 Proposed interactive real-time virtual fixture generation framework

2.1 General framework

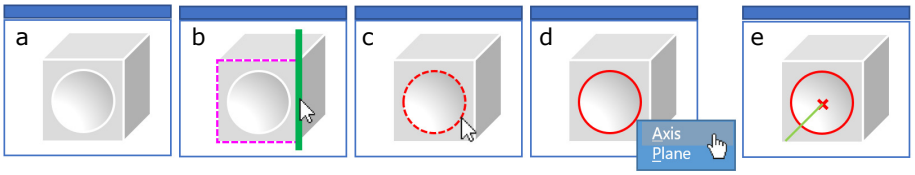

Fig. 1. Virtual fixture generation procedure for the peg in hole task. **a** the remote environment's objects are shown in the user interface. **b, c** the user hovers the mouse cursor over the object and its features are highlighted; if there are multiple features at the same cursor position, all of them will be displayed. **d** right mouse click on a desired feature reveals a pop-up menu that allows to generate the necessary fixture. **e** fixture is placed into the 3d environment for haptic rendering.

One of the main challenges of virtual fixture generation is that the virtual fixture has to be specified in the context of the teleoperation task. The context could be extracted automatically by setting logical rules and by providing 3D models prior to the actual teleoperation, but this might be tedious or even not possible if the environment is unstructured. In this paper, we propose to use human cognitive abilities to define the context manually from the two-dimensional projected view from the slave robot's camera and augmented reality GUI. The augmented reality GUI highlights a set of geometrical features of remote objects by graphically overlaying them in the camera view image in order to define the associated force field of virtual fixture.

In the proposed framework, a human operator can interactively select features (using a computer mouse or a touch screen) on the image coming from the depth camera. With the help of dedicated computer vision algorithms which identify geometrical features of remote objects and overlay them on the video stream from the remote environment, human operator can intuitively select features for assigning virtual fixture following the given teleoperation task context.

The process of fixture generation for the peg-in-hole task can be visualized as follows (Figure 1). First, operator observes the user interface where an image of the remote environment is displayed. Then, by hovering computer mouse, various features are highlighted, e.g. connected components, line segments, and circles. A virtual guidance that can be used for the peg in hole task is an axis of the hole. To find this axis, it is sufficient for the user to find the circular edge of the hole, and then the pose of the virtual fixture can be computed automatically by computing the center of the hole and normal axis of the center. All the necessary computations are done in the background, and the overall process is intuitive and fast to execute.

2.2 Interactive generation of virtual fixtures

This subsection describes how virtual fixtures are interactively generated based on the assigned features and their properties by a human operator.

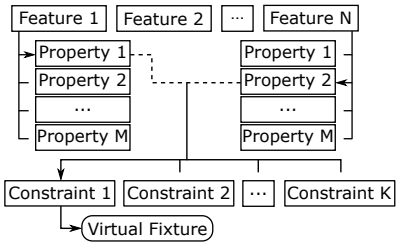

Fig. 2. The process of virtual fixture generation using features from 2D image: human operator selects *Feature 1* and *Property 1* of the tool in the user interface; then, he or she selects *Feature N* and *Property 2* describing the teleoperation target; finally, two properties are constrained together with *Constraint 1* which will define the virtual fixture. A *line with an arrow* shows the choice of the human operator, while *a line not touching an item* is the possibility of a choice.

Figure 2 introduces the process of interactive virtual fixture generation. Three components are used to generate virtual fixtures: a *feature* is a set of pixels highlighted in the user interface on mouse hover; a *property* is the corresponding 3D geometry of each feature, which is necessary to register the virtual fixture in 3D space; a *constraint* is a geometrical relation between two *properties*. First, human operator selects a feature of the tool (i.e., end effector or tool of the slave robot), and selects a property of the feature. Second, human operator selects a feature of the target object in remote environments, and selects the property of the chosen feature. Once two properties are selected, the operator can assign a constraint, which defines the geometrical relationship between two properties, and this is sufficient for the definition of the virtual fixture.

The diagram in Figure 3 shows the pipeline that computes features and their respective properties from RGB-d camera. Image features such as edges, line segments, and ellipses are detected in the image from the RGB camera and then presented to the user in the GUI. Pixels of the 2D features selected by the user are then transformed into 3D points using corresponding pixels of the depth image and sensor's intrinsic and extrinsic parameters. The user is able to choose a property that is the most useful for task-specific virtual fixture definition, like central point and normal to a surface composed of feature's points. Then, the user chooses an effect the fixture should produce: attraction or repulsion force. The pose of the fixture and its effect is transferred to the shared control system providing haptic assistance to the user through the master device.

Dedicated algorithms should be used for geometrical surfaces recognition. Circle/ellipse detection algorithm was developed for the teleoperation task pre-

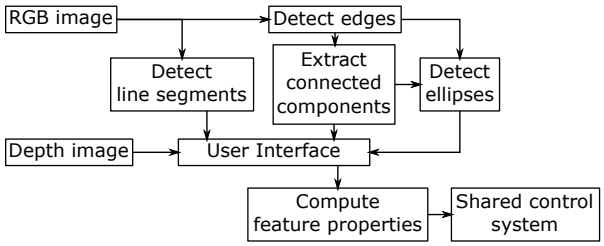

Fig. 3. Feature computation pipeline. Features and their properties from the RGB and Depth image streams are being computed in the real-time and stored in the User Interface. The human operator interacts with the User Interface to define a virtual fixture which is then added to the shared control system.

sented in this paper (peg-in-hole). This algorithm is described in details in the next subsection.

2.3 Detecting basic geometrical features: circle/ellipse

In order to provide an intuitive tool to human operator for assigning the virtual fixture, automatic feature and property detection algorithm running in the background is important. There are many different type of features and detection algorithms, but in this paper, we introduce a circle/ellipse detection algorithm only for briefly showing feasibility of the proposed framework.

The ellipse classification algorithm is described in Algorithm 1. The algorithm takes a set of 3D points as an input and tells if the points belong to an ellipse, or not. We use the following notations: $\mathbf{P}$ is a set of vectors or scalars, or a matrix; $\{\cdot\}$ is a set, $\boldsymbol{p}$ is a vector, p is a scalar, $\|\cdot\|$ is euclidean norm, $\overline{\cdot}$ is mean of a set.

The idea of the ellipse classification algorithm is to compute the distance d from the given set of points $\mathbf{P}$ of the connected component to the points of an ellipse $\mathbf{E}$ having the same major and minor axes obtained from principal component analysis (PCA) applied to $\mathbf{P}$. To apply PCA, we need to centralize the values of the pixels (steps 1 and 2); on step 3 obtain projection $\hat{\mathbf{P}}$ of $\mathbf{P}$ on principal component vectors. We can find minor and major axes of $\hat{\mathbf{P}}$ as minimum and maximum distance from points of $\hat{\mathbf{P}}$ to the origin. On steps 5 and 6 we then compute points of the ideal ellipse E matching angles and axes of $\hat{\mathbf{P}}$. On step 7 we compute distance d between points of $\mathbf{E}$ and $\hat{\mathbf{P}}$. The resulting distance value is then compared with the threshold ε. If $d < \varepsilon$ then we can say the points belong to an ellipse, otherwise they don't. We set the threshold value ε to be equal to 1 which means that the accumulated error from the image points to the corresponding ellipse should not exceed 1 pixel. In our usage experiments this algorithm is accurate enough as long as the edge detection algorithm extracts consistent edges.

Algorithm 1 Ellipse Classification Algorithm

Output:

Coordinates of points in a connected component: $\mathbf{P} = \{\boldsymbol{p}_i\},\ \mathbf{P} \subset \mathbb{R}^2\ i = 1, 2, \ldots, n$, n is a matrix of points

Threshold : ε

Ensure:

The classification result: true or false

1: $\boldsymbol{c} = \sum_i^n \boldsymbol{p}_i / n$ ▷ compute center of $\mathbf{P}$

2: $\tilde{\mathbf{P}} = \mathbf{P} - \boldsymbol{c} = \{\boldsymbol{p}_i - \boldsymbol{c}\}$ ▷ shift $\mathbf{P}$ towards the center $\boldsymbol{c}$

3: $\hat{\mathbf{P}} = \tilde{\mathbf{P}}\mathbf{W}$ ▷ Apply PCA to $\tilde{\mathbf{P}}$; $\mathbf{W}$ is a n-by-n matrix whose columns are eigenvalues of $\tilde{\mathbf{P}}^T\tilde{\mathbf{P}}$

4: $a = \min_i \|\hat{\boldsymbol{p}}_{\boldsymbol{i}}\|\ ,\ b = \max_i \|\hat{\boldsymbol{p}}_{\boldsymbol{i}}\|$ ▷ find major a and minor b axes of $\hat{\mathbf{P}}$

5: $\boldsymbol{\Phi} = \{\phi_i\} = \{\arctan 2(\frac{\hat{p}_{iy}}{a}, \frac{\hat{p}_{ix}}{b})\}$ ▷ compute polar angles of $\hat{\mathbf{P}}$

6: $\mathbf{E} = \{\boldsymbol{e}_i\} = \{(a\cos\phi_i, b\sin\phi_i)\}$ ▷ compute points of ellipse with axes a, b for angles $\boldsymbol{\Phi}$

7: $d = \overline{\left\|\mathbf{E} - \hat{\mathbf{P}}\right\|} = \frac{\sum_i^n \|\boldsymbol{e}_i - \hat{\boldsymbol{p}}_i\|}{n}$ ▷ compute mean distance d of ideal ellipse points $\mathbf{E}$ and projected values $\hat{\mathbf{P}}$

8: **return** $d < \varepsilon$ ▷ if holds true, then points in P belong to an ellipse, otherwise not

3 Experiment

3.1 Experimental setup

Our bilateral teleoperation system was composed of a UR5 universal robotic manipulator (slave) equipped with a Delta force/torque sensor and CMU Cam 5 video camera at the end effector; Omega 6 haptic interface (master); Microsoft Kinect RGB-d camera installed at the slave side; control computer with the proposed interactive virtual fixture definition GUI. The overall experimental setup can be seen in Figure 4.

The peg-in-hole task was evaluated during the experiment. The 3D printed peg was rigidly attached to the slave's end-effector. The diameter of the peg was 30 mm and diameter of the hole was 30.5 mm. The setup is based in the KoreaTech Biorobotics lab. Slave and master subsystems were located away from each other and were connected through an Ethernet network and it was assumed to be time delay free. We have also evaluated that there was no packet loss within the network. Force feedback from the slave side was based on direct force measurements from the end-effector's force-torque sensors. Direct position-to-position mapping was used to control the slave robot with the master device. Stability of the bilateral teleoperator was guaranteed by the PO/PC approach (passivity observer and passivity controller [8]). The controller was running at 4 kHz sampling rate.

The ArUco augmented reality marker [7] was used to localize the slave manipulator in the Kinect camera's frame of reference. As a result, the teleoperation controller and virtual fixture definition GUI were operating in the same frame of reference.

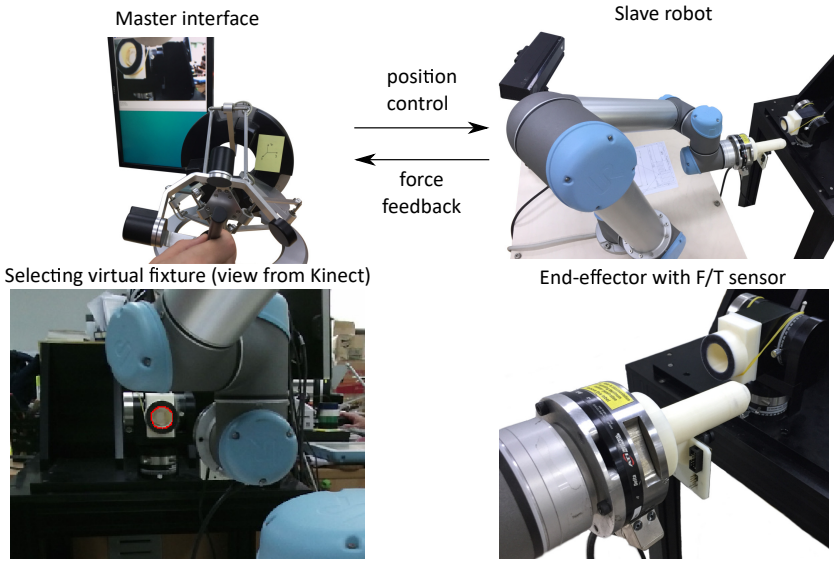

Fig. 4. Experimental setup for bilateral teleoperation with virtual fixtures

3.2 Generation of virtual fixtures

The process of the fixture generation for the peg-in-hole task with the proposed GUI consists of 9 steps (as illustrated in Figure 5):

1. The human operator enables the *feature selection mode*; in this mode hovering the mouse over the camera view reveals *features*.
2. The human operator finds the necessary feature
3. Clicking on the found feature selects its type (*ellipse*).
4. The ellipse appears in the *detected features* section (the human operator can select multiple features before proceeding to next step.
5. Clicking on *apply* combines the selected features into one.
6. The combined feature appears in the *combined features* menu; the human operator clicks on the *combined feature*,
7. The geometrical *properties* of the feature appear in a drop-down menu.
8. Clicking on the necessary *property* of the *combined feature*, the user selects a *geometrical constraint*, e.g. *Align*.
9. The *align with the hole's axis* feature is displayed in the user interface and placed in the virtual environment to provide guidance to the human operator.

Current implementation supports *features*: connected components, line segments, ellipses. These features on a 2d image, projected to the depth image from the Kinect sensor, allow to restore 3d positions of the image points; the 3d positions are then used to compute the center of the set of points, normal to the plane fitting the set of points, and spatial orientation of asymmetric planar

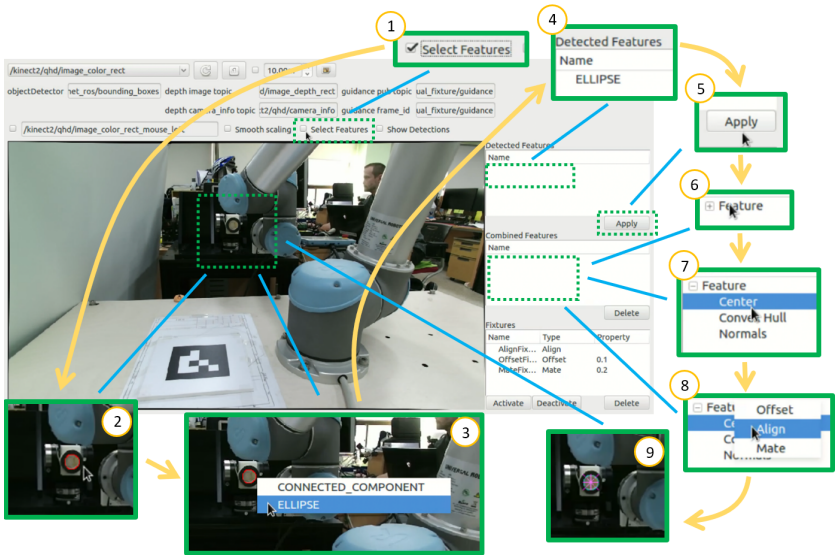

Fig. 5. User interface with description of the virtual fixture generation process.

geometrical objects. The fixtures that can be generated using these features and their properties are *offset, align*, and *mate*.

The user interface supports combining of multiple features into one (Figure 5 *3-5*). For example, if the edges of the circle for the peg-in-hole task cannot be detected using a single edge contour, then multiple pieces of the contour can be selected.

The whole procedure of the fixture generation consists of 9 actions that are performed by moving the computer mouse and clicking. As most of the features are computed and highlighted automatically, the user doesn't need to perform precise mouse movements. During the experiments, it has been shown that subjects got used to the user interface quickly, and the process of fixture generation takes very short time.

3.3 Protocol and participants

The goal of the study was to demonstrate that the proposed virtual fixture definition method can be efficiently used in a simple teleoperation scenario. The task of a participant was to complete the peg-in-hole task as fast and as accurate as possible based on limited visual feedback from the end-effector's camera and force feedback from the force-torque sensor. Through the end-effector's camera the participants could see the tip of the peg and the target environment. Additionally, the participants had access to the virtual fixture GUI, so that they could define a guidance fixture for the hole's position in the peg-in-hole task. Two conditions were tested: 1) teleoperation without virtual fixtures and 2) teleoperation with virtual fixtures. Ten participants took part in the evaluation tests (all

male, age 23-35), nine of them were right-handed, one - left-handed. They were instructed how to perform the peg-in-hole task as fast as possible and were given an opportunity to perform several familiarization trials with and without haptic guidance. This was then followed by a set of experimental trials with guidance (5 trials) and without guidance (5 trials). The trials with and without guidance were mixed in order to prevent the adaptation effects. During the experiments, the participants could not see the slave robot as it was intentionally hidden from the field of view.

For each trial, we have recorded the master's and slave's position, measured slave's end-effector force and haptic guidance force if it was used. Time to the first contact with the hole, total completion time, smoothness of master and slave movements, and magnitude of interaction forces were used as performance indicators.

4 Results

All human-subjects were able to complete the trials successfully. Figure 6 presents the total completion time and time to the first contact for all subjects with and without haptic guidance for 10 participants (mean and standard deviation of measured time in all trials). Virtual fixture selection time was not included in the total task completion time. For all participants, average task completion time with a virtual guidance was 15.22 ± 5.55 s (mean±standard deviation) which is less when the virtual guidance was not used 21.09 ± 6.10 s. Time to the first contact was also reduced for all participants when haptic guidance was used: 12.44 ± 3.57 s against 15.61 ± 4.23 s.

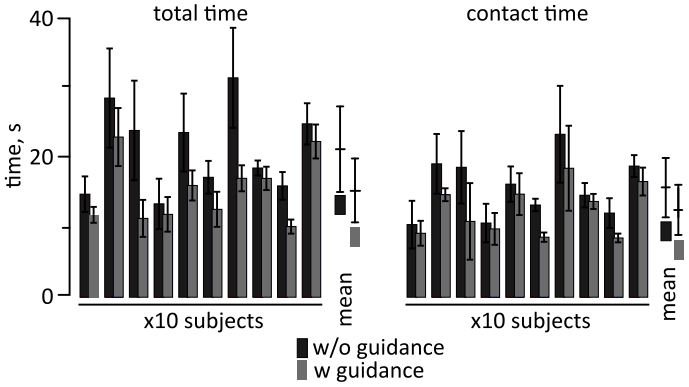

Fig. 6. Results for task completion time per subject with and without virtual fixtures.

Figure 7 presents time series plot with and without guidance for a typical subject (master/slave position, force feedback from the force-torque sensor, and force rendered at master side). Figures 7a–c show the plots for master/slave

position, end-effector force measurements, and rendered force at the master side for the case when no virtual fixtures were used. Figures 7d-f show the plots for master/slave position, end-effector force measurements, and rendered force at the slave side for the case when the virtual guidance was enabled. The positions of master and slave devices (Figure 7a,d) drop to 0 occasionally, this happens when the human operator releases the button at the master device to perform indexing. We can observe that the magnitudes of the measured end-effector force when the peg was inside the hole were smaller for the case when the virtual guidance was enabled. The motion trajectory with virtual guidance enabled was also smoother. These results allow to conclude that the virtual fixture generated in real-time by the operator provided a better quality of motion; it enabled the human operator to move more safely and precisely. Finally, both plots demonstrate that the task execution time was smaller for enabled virtual guidance as it is also visualized in the Figure 6.

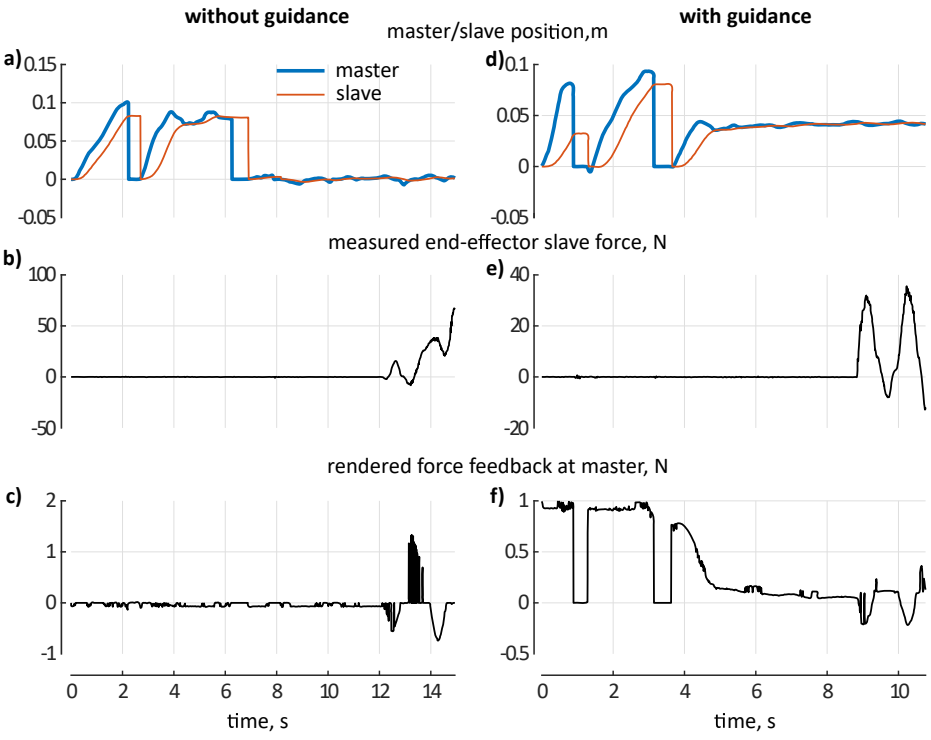

Fig. 7. Position and force profile without (**a,b,c**) and with (**d,e,f**) virtual guidance. Plots: **a,d** - the master's and slave's positions; **b,e** - measured force at the slave's end-effector; **c,f** - rendered force at the master.

5 Conclusion and future works

This paper presents a novel interactive virtual fixture generation method for shared teleoperation in unstructured remote environments. The proposed method allows human operator to easily assign and register virtual fixtures on the remote target objects on-the-fly with only several mouse button clicks. The easiness of the virtual fixture generation as well as the effectiveness of the real-time generated virtual fixture was tested with a peg-in-hole task. The human subject study shows improved performance of the peg-in-hole teleoperation task with real-time generated virtual fixture.

As a future work, we are currently generalizing the proposed framework including diverse type of features, properties and constraints, and planning to test the performance and effectiveness of the proposed method with more complicated unstructured teleoperation tasks.

Acknowledgements

This research is supported by the project "Toward the Next Generation of Robotic Humanitarian Assistance and Disaster Relief: Fundamental Enabling Technologies (10069072)" and by the project (Development of core teleoperation technologies for maintaining and repairing tasks in nuclear power plants) funded by the Ministry of Trade, Industry & Energy of S. Korea.
I. Farkhatdinov was partially supported by the UK EPSRC in the framework of the National Centre for Nuclear Robotics project (EP/R02572X/1).

References

1. Aarno, D., Ekvall, S., Kragic, D.: Adaptive virtual fixtures for machine-assisted teleoperation tasks. In: Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on. pp. 1139–1144.
2. Abbott, J.J., Panadda, M., Allison, O., Marayong, P., Okamura, A.M.: Haptic Virtual Fixtures for Robot-Assisted Manipulation. In: Robotics Research, 49–64.
3. Bettini, A., Lang, S., Okamura, A., Hager, G.: Vision assisted control for manipulation using virtual fixtures: experiments at macro and micro scales. In: Proc. 2002 IEEE International Conference on Robotics and Automation, vol. 4, pp. 3354–3361.
4. Bowyer, S.A., Davies, B.L., Rodriguez Y Baena, F.: Active constraints/virtual fixtures: A survey. IEEE Transactions on Robotics **30**(1), 138–157 (2014).
5. Farkhatdinov, I., Ryu, J.H.: Improving mobile robot bilateral teleoperation by introducing variable force feedback gain. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 5812–5817 (Oct 2010).
6. Farkhatdinov, I., Ryu, J.H., Poduraev, J.: A user study of command strategies for mobile robot teleoperation. Intelligent Service Robotics **2**(2), 95–104 (2009)
7. Garrido-Jurado, S., noz Salinas, R.M., Madrid-Cuevas, F., Marín-Jiménez, M.: Automatic generation and detection of highly reliable fiducial markers under occlusion. Pattern Recognition **47**(6), 2280 – 2292 (2014).
8. Hannaford, B., Ryu, J.H.: Time-domain passivity control of haptic interfaces. IEEE T. on Robotics and Automation **18**(1), 1–10 (2002).

9. Jiang, Z., Liu, Y., Liu, H., Zou, J.: Flexible virtual fixture enhanced by vision and haptics for unstructured environment teleoperation. In: 2013 IEEE International Conference on Robotics and Biomimetics (ROBIO). pp. 2643–2648.
10. Li, M., Taylor, R.H.: Spatial motion constraints in medical robot using virtual fixtures generated by anatomy. In: Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on. vol. 2, pp. 1270–1275.
11. Lin, H.C., Mills, K., Kazanzides, P., Hager, G.D., Marayong, P., Okamura, A.M., Karam, R.: Portability and applicability of virtual fixtures across medical and manufacturing tasks. In: Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on. pp. 225–230.
12. Ming Li, Kapoor, A., Taylor, R. A constrained optimization approach to virtual fixtures. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems 2015, 1408–1413.
13. Quintero, C.P., Dehghan, M., Ramirez, O., Ang, M.H., Jagersand, M.: Flexible virtual fixture interface for path specification in tele-manipulation. In: 2017 IEEE Int. Conference on Robotics and Automation (ICRA). pp. 5363–5368 (May 2017).
14. Rosenberg, L.: Virtual fixtures: Perceptual tools for telerobotic manipulation. In: Proc. of IEEE Virtual Reality Annual International Symposium. pp. 76–82.
15. Rosenberg, L.B.: The Use of Virtual Fixtures as Perceptual Overlays to Enhance Operator Performance in Remote Environments. (1992)
16. Ryden, F., Chizeck, H.J.: Forbidden-region virtual fixtures from streaming point clouds: Remotely touching and protecting a beating heart. IEEE International Conference on Intelligent Robots and Systems pp. 3308–3313 (2012).
17. Selvaggio, M., Chen, F., Gao, B., Notomista, G., Trapani, F., Caldwell, D.: Vision based virtual fixture generation for teleoperated robotic manipulation. In: 2016 Int. Conference on Advanced Robotics and Mechatronics, pp. 190–195.
18. Selvaggio, M., Member, S., Notomista, G., Member, S., Chen, F., Gao, B., Trapani, F., Caldwell, D.: Enhancing bilateral teleoperation using camera-based online virtual fixtures generation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 1483–1488.